UNIVERSIDADE DE SÃO PAULO ESCOLA POLITÉCNICA
DEPARTAMENTO DE ENGENHARIA ELÉTRICA

SISTEMA DE DETECÇÃO E CLASSIFICAÇÃO DE NOTÍCIAS FALSAS

Daniela Yumi Sekiguchi
Letícia Saori Tsuji
Nicolas Gobbi Gonçalves

São Paulo - SP
Dezembro de 2021

Catalogação-na-publicação

Sekiguchi, Daniela
SISTEMA DE DETECÇÃO E CLASSIFICAÇÃO DE NOTÍCIAS FALSAS / D. Sekiguchi, L. Tsuji, N. Gonçalves -- São Paulo, 2021.
41 p.

Trabalho de Formatura - Escola Politécnica da Universidade de São Paulo. Departamento de Engenharia de Telecomunicações e Controle.

1.Fake news 2.Aprendizado computacional 3.Computação em nuvem 4.Análise de dados I.Universidade de São Paulo. Escola Politécnica. Departamento de Engenharia de Telecomunicações e Controle II.t. III.Tsuji, Letícia IV.Gonçalves, Nicolas

UNIVERSIDADE DE SÃO PAULO ESCOLA POLITÉCNICA
DEPARTAMENTO DE ENGENHARIA ELÉTRICA

SISTEMA DE DETECÇÃO E CLASSIFICAÇÃO DE NOTÍCIAS FALSAS

Trabalho de formatura apresentado à Escola Politécnica da Universidade de São Paulo para obtenção do título de Graduação em Engenharia.

Daniela Yumi Sekiguchi
Letícia Saori Tsuji
Nicolas Gobbi Gonçalves

Orientador: Felipe Miguel Pait

Área de Concentração:
Engenharia Elétrica

São Paulo - SP

Dezembro de 2021

RESUMO

A facilidade cada vez maior na difusão de conteúdo vem acompanhada de uma politização da informação, que torna mais comum a distorção da verdade e difusão de conteúdo enganoso, conhecido como *fake news* ou notícias falsas. Para combater essa difusão, foi desenvolvido um projeto que consiste na construção de um modelo de aprendizado de máquina, que detecta notícias falsas em português brasileiro. Visando atingir esse objetivo, foi criada uma base de dados com notícias atuais, falsas e verdadeiras, dois modelos treinados de aprendizado de máquina (um para parâmetros semânticos e um para sintáticos) e um serviço para oferecer, de modo online, os resultados dos modelos treinados dada, uma notícia. Foram obtidos resultados com acurácia e precisão acima de 95% para os modelos em testes isolados, porém alguns testes mostraram viés relacionado ao tamanho das notícias.

ABSTRACT

With growth in information dissemination comes an increase in information politicization, which produces distortion of truth via the spread of fake news. In order to diminish this spread, a project consisting in the construction of a machine learning brazilian portuguese fake news detection model was developed. To achieve the original goal, a process of producing a corpus of true and false brazilian news was executed, which was later used to train the machine learning models - semantic and syntactic techniques were used. Also, to expose the functionality, a website which allowed access to the model was developed. Regarding the model, accuracy and precision results over 95% were obtained in isolated tests, although some users in real tests have pointed out that the algorithm seemed biased by the size of the news article.

LISTA DE FIGURAS

Figura 1: Exemplo de classificação com K = 3. No caso, a classe atribuída seria azul (2 vizinhos azuis, 1 amarelo, maioria azul)..... 06
Figura 2: Esquerda: Linhas representam hiperplanos para 2 parâmetros, círculos e quadrados são as 2 classes. Direita: Linha contínua representa hiperplano ótimo para a base, as observações preenchidas representam os *support vectors*, linhas pontilhadas são a margem (máxima entre os possíveis hiperplanos)..... 07
Figura 3: Classificação com SVC - *kernel* radial..... 07
Figura 4: Árvore simples de decisão para caso a pessoa sobreviva ou morra..... 08
Figura 5: Exemplo de Perceptron..... 09
Figura 6: Partes de um nodo 09
Figura 7: Histograma de número de tokens por classificação 15
Figura 8: *Polarity* por *num_tokens*..... *16*
Figura 9: Histograma de polaridade por classificação..... 16
Figura 10: Gráfico de dispersão da iminência..... 18
Figura 11: Gráfico em blocos de *non-immediacy*..... 18
Figura 12: Histograma do parâmetro *average_sentence_len*..... 19
Figura 13: CloudWords das notícias..... 20
Figura 14: Métricas para diferentes profundidades de Árvore de Decisão..... 24

LISTA DE TABELAS

Tabela 1: Exemplo de *tokenização* 03
Tabela 2: Definição das classes do *POS-tagging*..... 04
Tabela 3: Exemplo de *POS-tagging*..... 04
Tabela 4: Exemplo de *stem* de palavras..... 05
Tabela 5: Exemplo de *stopwords*..... 05
Tabela 6: Exemplo de *bag of words*..... 05
Tabela 7: Matriz de Confusão 11
Tabela 8: Domínios de origem das notícias do Fake.br Corpus 12
Tabela 9: Domínios utilizados para obter notícias verdadeiras pelo *crawler* 13
Tabela 10: Domínios utilizados para obter notícias verdadeiras pelo *crawler* 13
Tabela 11: Dados contidos em cada pool utilizado 22
Tabela 12: Quantidades de notícias retiradas por site, notícias verdadeiras na tabela da esquerda e notícias falsas na tabela da direita, para a construção da base 26
Tabela 13: quantidade de notícias verdadeiras e falsas em cada base de dados 26
Tabela 14: Melhores valores de k para diferentes combinações de pool e base para os modelos de KNN e Random Forest. 27
Tabela 15: Melhores resultados obtidos para acurácia e precisão dos modelos criados anteriormente após refino..... 27
Tabela 16: Melhores resultados para cada tipo de modelo para *bag of words* 28
Tabela 17: Melhores resultados para cada tipo de modelo para *bag of words* alternativo..... 29

SUMÁRIO

LISTA DE TABELAS

LISTA DE FIGURAS

1.INTRODUÇÃO..... 01

1.1.Motivação e Contextualização..... 01

1.2.Objetivo..... 01

2.ASPECTOS CONCEITUAIS/BASE TEÓRICA..... 02

2.1.Fake News..... 02

2.2.Processamento de Linguagem Natural..... 03

2.3.Aprendizado de Máquina..... 06

2.3.1.Modelos..... 06

2.3.2.Avaliação dos modelos..... 10

3.METODOLOGIA..... 12

3.1.Base de dados..... 12

3.2. Extração e Análise dos Dados..... 14

3.2.1.Parâmetros sintáticos..... 14

3.2.2.Parâmetros semânticos (Bag of words)..... 20

3.3.Modelagem..... 21

3.3.1. Pré-processamento..... 21

3.3.2. Montagem do modelo..... 22

3.3.3. Avaliação dos modelos..... 23

3.4.Construção do serviço web..... 24

4.RESULTADOS..... 26

4.1.Base de dados..... 26

4.2.Análise dos Modelos..... 26

4.3.Construção do serviço..... 29

5.DISCUSSÃO..... 30

6.CONSIDERAÇÕES FINAIS..... 31

6.1.Conclusões do Projeto de Formatura..... 31

6.2.Trabalhos Futuros..... 31

LISTA DE REFERÊNCIAS..... 32
ANEXOS..... 35

1.INTRODUÇÃO

1.1.Motivação e Contextualização

A facilidade cada vez maior na difusão de conteúdo vem acompanhada de uma politização da informação, que torna mais comum a distorção da verdade e difusão de conteúdo enganoso, criado ou por acidente, ou com o propósito de desinformar. Esse tipo de conteúdo, em contato com agentes de forte disseminação, é conhecido como *fake news* ou notícias falsas [1].

O tema vem ganhando relevância por trazer implicações em diferentes setores, como política e saúde [2], com destaque para o ano da pandemia da COVID no Brasil, durante o qual a incerteza em torno da doença abriu espaço para surgimento de desinformação fantasiosa ou sem qualquer tipo de embasamento científico, muitas vezes com fim de favorecer alguma entidade ou ideal político. O problema cresceu tanto nos últimos anos que levou ao surgimento de diversas agências dedicadas à averiguação de fatos, e, recentemente, até à instalação de uma Comissão Parlamentar de Inquérito que busca investigar as maiores fontes desses tipos de notícias.

A detecção de *fake news* por métodos de aprendizado de máquina já é um tema cada vez mais recorrente na literatura. Para o português brasileiro, há projetos que atingem métricas perto de 97% [26].

1.2.Objetivo

O objetivo principal deste trabalho consiste na construção de um modelo de aprendizado de máquina que detecta notícias falsas em português brasileiro. Essa meta pode ser dividida em cinco etapas: estudo sobre o tema, extração das características do banco de dados, análise e tratamento dos dados, ajuste de modelos para classificação de notícias, avaliação e escolha. Os objetivos secundários são: expandir o banco de dados utilizado e criar um website de classificação de notícias.

2.ASPECTOS CONCEITUAIS/BASE TEÓRICA

2.1.Fake news

O termo *fake news* cotidianamente significa conteúdo enganoso, encontrado principalmente na mídia. No entanto, para enganar alguém, não é necessária intenção maliciosa nem mesmo informação falsa. Dependendo de como essa informação é disseminada, apresentada e vários outros aspectos, o receptor pode montar uma compreensão errônea da realidade. Assim, levando em consideração as diferentes características exibidas, torna-se difícil definir o que é *fake news*.

Em textos, há aqueles que utilizam *fake news* como desinformação de alto alcance, disseminada principalmente por redes sociais [1], ou simplesmente consideram apenas artigos com conteúdo fabricado [3]. Já outros procuram dividir e classificar o termo em diferentes partes. Claire Wardle [2] separa-os em sete tipos: sátira (potencial para enganar, sem intenção maliciosa), conexão falsa (primeira impressão diferente do conteúdo real), conteúdo enganador (uso de informação de forma enganosa para prejudicar algo ou alguém), contexto falso (uso de informação válida em contexto falso), conteúdo impostor (falsificação de fontes verídicas), conteúdo manipulado (manipulação de informação para enganar leitor) e conteúdo fabricado (informação totalmente falsa com intenção maliciosa do criador). Essa divisão é apenas uma das formas de definir *fake news*.

- *Fact-checking*

O termo *fact-checking* refere-se à ação de checar a veracidade de uma notícia. Esse ato é várias vezes relacionado a *fake news*, devido ao contexto. No Brasil, há várias e diferentes agências realizando esse trabalho manualmente, entre elas a Lupa e a Aos Fatos. A Agência Pública também era uma delas, mas seu projeto, Truco, foi encerrado em 2018. Todas utilizam uma metodologia parecida, em que trechos específicos da informação são verificados por especialistas. Essas frases são escolhidas pela chance de conterem informação falsa, dependendo de sua relevância ou impacto.

O método, no entanto, não é à prova de falhas. Chloe Lim [4] afirma que essa metodologia não pode ser considerada científica, e testa que cerca de 20% das análises de duas grandes agências estadunidenses tiveram classificações conflituosas.

Além das agências de *fact-checking*, há também um serviço criado por pesquisadores da USP e da UFSCar, chamado Fake Check. O aplicativo classifica notícias utilizando aprendizagem de máquina, com base apenas em seu texto. De acordo com seu artigo, o projeto demonstra uma acurácia de 90%[5].

- Detecção de *fake news*

Para combater as *fake news*, profissionais da área da informação recomendam checar fontes confiáveis e desconfiar de notícias polêmicas. O site do Diretório Acadêmico de Gestão de Informação (https://sites.ufpe.br/dagi/2020/07/05/como-identificar-fake-news/) também destaca em *fake news* o excesso de pontuação, erros gramaticais e palavras em caixa alta.

2.2.Processamento de Linguagem Natural

Natural Language Processing (NLP) é uma área estudada desde a década de 50, quando Alan Turing propôs o "Teste de Turing", buscando uma forma de verificar se uma máquina podia pensar [23]. Hoje em dia, com o avanço da computação, NLP está estritamente associado a métodos de *machine learning*, os quais serão explorados neste trabalho. Assim, o NLP já agrega diversos termos e ferramentas, facilitando seu uso e diversificando suas aplicações no aprendizado de máquina.

- *Token*

O termo *token* refere-se às partículas de um texto como palavras, números e pontuações, como exemplificado na tabela X. A extração de *tokens* (*tokenização*) ocorre por meio de um *tokenizer*.

Texto inicial	***Tokens* do texto**
"O cachorro comeu o meu TCC."	"O", "cachorro", "comeu", "o", "meu", "TCC", "."

Tabela 1: Exemplo de *tokenização*.

- *Part-of-speech* (*POS-tagging*)

É chamada de *part-of-speech* ou *POS-tagging* uma determinada classificação das palavras levando em consideração a relação entre elas no texto e suas posições na

frase, como exemplificado na tabela 3. Essa classificação é baseada na divisão gramatical da língua inglesa e está listada na tabela 2.

Sigla (*POS-tagging*)	Definição
ADJ	*Adjective*
ADP	*Adposition*
ADV	*Adverb*
AUX	*Auxiliary*
CCONJ	*Coordinating Conjunction*
DET	*Determiner*
INTJ	*Interjection*
NOUN	*Noun*
NUM	*Number*
PRON	*Pronoun*
PROPN	*Proper Noun*
PUNCT	*Punctuation*
SCONJ	*Subordinating Conjunction*
VERB	*Verb*

Tabela 2: Definição das classes do *POS-tagging*.

Texto inicial	*POS-tagging* das palavras do texto
“O cachorro comeu o meu TCC.”	“O”: DET “cachorro”: NOUN “comeu”: VERB “meu”: DET “TCC”: PROPN “.”: PUNCT

Tabela 3: Exemplo de *POS-tagging*.

- *Stem*

O termo *stem* representa a parte inicial das palavras, sem sufixos, o que permite ignorar conjugações, gênero e número das palavras, como exemplificado na tabela 4. A ferramenta *stemmer* extrai o *stem*.

Palavras	***Stem* das palavras**
“cachorro”, “cachorros”, “cachorrinha”	“cachorr”
“comeu”, “come”, “comia”	“com”

Tabela 4: Exemplo de *stem* de palavras.

- *StopWords*

São chamadas de *stopwords* as palavras que não agregam muito significado à frase, podendo ser retiradas sem muita perda de sentido. As *stopwords* variam de caso para caso, porém são recorrentes as palavras listadas abaixo na tabela 5.

'de', 'a', 'o', 'que', 'e', 'é', 'do', 'da', 'em', 'um', 'para', 'com', 'não', 'uma', 'os', 'no', 'se', 'na', 'por', 'mais', 'as', 'dos', 'como', 'mas', 'ao', 'ele', 'das', 'à', 'seu', 'sua', 'ou', 'quando', 'muito', 'nós', 'já', 'eu', 'também', 'só', 'pelo', 'pela', 'até', 'isso', 'ela', 'entre'

Tabela 5: Exemplo de *stopwords.*

- *Bag of words*

O *bag of words* é um método de interpretação que utiliza apenas a frequência das palavras dentro do texto, ignorando suas posições. Assim, um texto pode ser representado a partir de um vetor, no qual cada dimensão representa uma palavra de um dicionário e o valor nela é a frequência em que a palavra aparece no texto, como exemplificado na tabela 6.

Texto inicial	***Bag of words* do texto**
“O cachorro comeu o meu TCC”.	“cachorro”: 1 “comeu”: 1 “meu”: 1 “o”: 2 “tcc”: 1

Tabela 6: Exemplo de *bag of words*.

- Análise de Sentimento

A análise de sentimentos é um dos desafios de NLP, que busca extrair elementos subjetivos, como opiniões e emoções, de textos. Para o projeto, destaca-se o conceito de polaridade, em que as palavras podem ter um valor positivo, negativo ou neutro emotivamente. [6] O cálculo da polaridade pode ser feito com base em um

léxico, que relaciona uma palavra a um certo valor (polaridade da palavra). No caso, a polaridade total do texto foi feita pela soma de todas as polaridades encontradas.

2.3.Aprendizado de Máquina

O aprendizado de máquina é uma tecnologia relativamente nova, com base em probabilidade, utilizado principalmente para previsão ou classificação de comportamentos. No caso, ele será utilizado para a classificação das notícias em verdadeiras ou falsas. Essa classificação é feita por um modelo, que decide o resultado a partir de características (*features*) extraídas do texto.

2.3.1.Modelos

Foram utilizados os modelos mais tradicionais de classificadores, sendo eles: KNN (K-nearest neighbors ou "K vizinhos mais próximos"), SVC (Support Vector Classification), Random Forest, Perceptron (Redes Neurais), Naive Bayes e Logistic Regression (Regressão Logística). Esses modelos serão discutidos em detalhe a seguir.

- KNN

O KNN (*K-nearest neighbors*) é um algoritmo simples e intuitivo de aprendizado de máquina. Para classificação, ele assume que observações de mesma classe se mantém próximas, assim a classe de um evento qualquer depende apenas da classe majoritária de seus K vizinhos, sendo K um inteiro menor ou igual ao número total de observações, como visto na figura 1. A distância entre as observações de uma base pode ser dada de diversas formas, sendo uma das mais conhecidas a distância Euclidiana.[7]

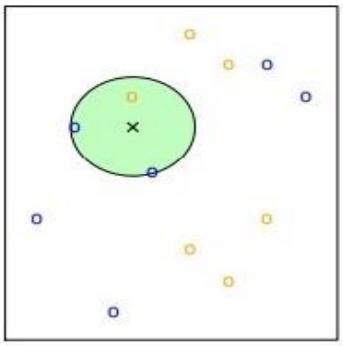

Figura 1: Exemplo de classificação com K = 3. No caso, a classe atribuída seria azul (2 vizinhos azuis, 1 amarelo, maioria azul).
Retirada de James et al. (2013) [13].

- SVC

O modelo por SVC utiliza hiperplanos para a classificação de uma observação. Hiperplanos são subespaços de dimensão n-1, sendo n a quantidade de parâmetros utilizados, que dividem as classes no espaço de dimensão n [8]. O classificador escolhido é aquele que melhor divide as classes e possui maior margem entre si e os *support vectors* (pontos que influenciam o hiperplano).

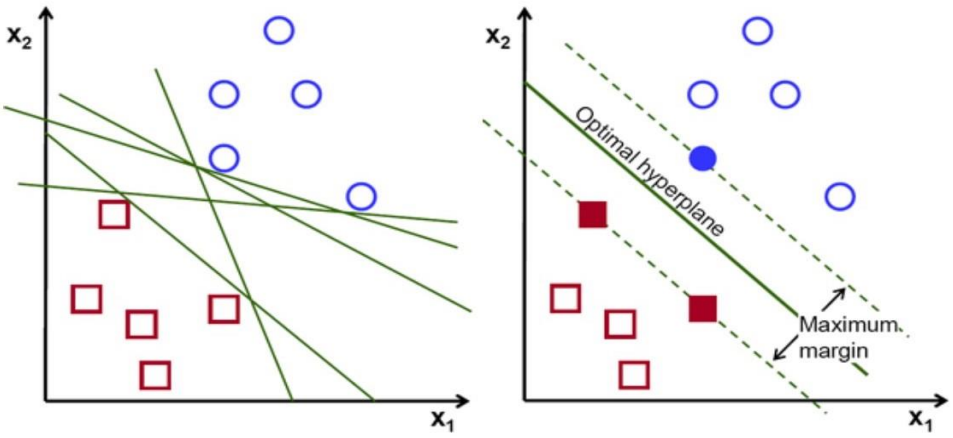

Figura 2: Esquerda: Linhas representam hiperplanos para 2 parâmetros, círculos e quadrados são as 2 classes. Direita: Linha contínua representa hiperplano ótimo para a base, as observações preenchidas representam os *support vectors*, linhas pontilhadas são a margem (máxima entre os possíveis hiperplanos).
Retirada de PAPANASTASIOU (2018) [3].

O exemplo da figura 2 é linear. Para casos não lineares, utiliza-se o *kernel*, que generaliza o produto interno do sistema, possibilitando assim uma flexibilização da barreira de decisão, como ocorre na figura 3.

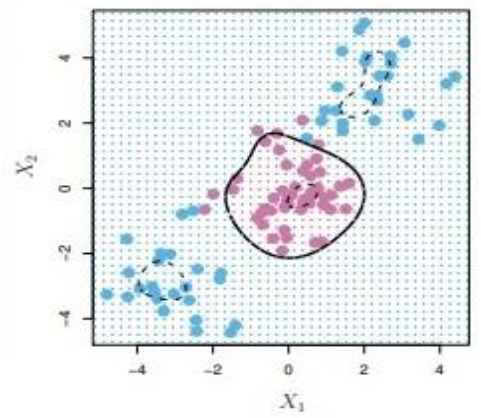

Figura 3: Classificação com SVC - *kernel* radial.
Retirada de FREITAS (2013) [6].

- Random Forest

A árvore de decisão (*decision tree*) é um método de classificação, cujo "raciocínio" é facilmente explicável, já que o modelo é baseado em uma árvore de regras. A partir do ponto inicial (raíz da árvore), o algoritmo decide qual o próximo ponto (nodo) de acordo com o valor de determinado parâmetro da observação (regra), até chegar em uma classificação (folhas). A figura 4 ilustra um exemplo de modelo.[9]

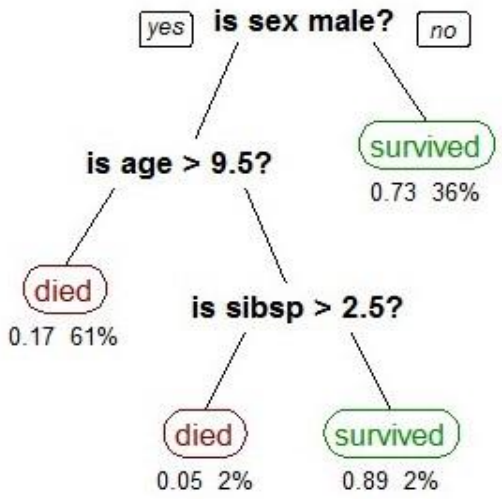

Figura 4: Árvore simples de decisão para caso a pessoa sobreviva ou morra.
Retirado de GUPTA 2017 [9].

Uma *random forest* basicamente é formada por várias árvores de decisão, e sua saída é decidida pela maioria. Diferentemente da árvore de decisão em si, as árvores em *random forests* utilizam parâmetros escolhidos de forma aleatória e limitada, e são treinadas cada uma com diferentes partes da base de treino. Assim, alguns problemas presentes nas árvores de decisão não reaparecem em *random forests*, como o *overfitting.*

- Perceptron

Redes neurais (*neural networks*) são um dos métodos mais famosos de aprendizado profundo. Inspiradas no funcionamento de neurônios, elas são formadas por unidades que se comunicam de forma unidirecional [10]. Essas redes contêm normalmente duas partes: uma linear, lidando com a somatória das entradas multiplicada pelos pesos, e outra não linear, normalmente a aplicação de uma função

não linear (sigmoide, por exemplo) na saída da parte linear (figura 6). O Perceptron é um tipo de rede neural com apenas uma camada escondida, como na figura 5.

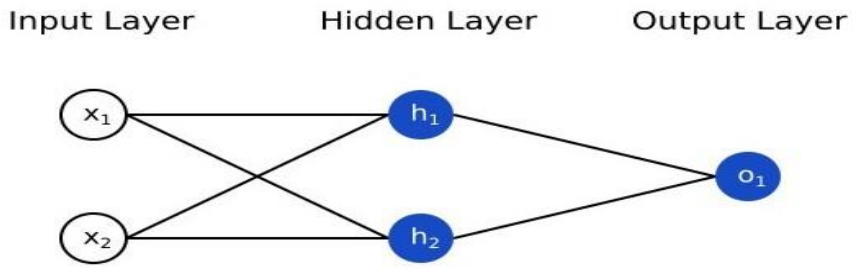

Figura 5: Exemplo de Perceptron
Retirado de FRETAS (2013) [6]

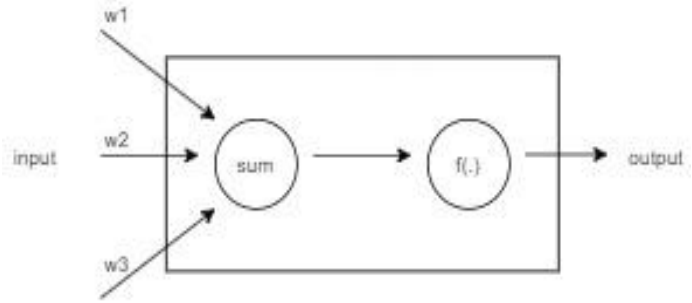

Figura 6: Partes de um nodo

- Naive Bayes

Naive-Bayes é um classificador probabilístico baseado na aplicação do Teorema de Bayes, que descreve a probabilidade de ocorrência de um evento, baseado no conhecimento das condições que levam àquele evento [12]. Esse método de classificação é uma das formas mais simples de aplicação dos modelos de rede Bayesiana, e pressupõe forte independência entre as *features.* Dessa forma, a probabilidade de um evento ocorrer é:

$$P(Y = c_k | X = x) = \frac{P(X = x | Y = c_k) P(Y = c_k)}{P(X = x)}$$

$$P(X = x | Y = c_k) = \Pi_i P(X = x_i | Y = c_k)$$

$$P(X = x) = \sum_k P(X = x | Y = c_k)$$

Sendo x o evento ou a observação que se quer classificar, x_i cada *feature* do evento e c_k cada classe possível.

- Logistic Regression

A regressão logística é outro classificador probabilístico. No caso, a probabilidade de, dado um evento x, a classe ser c é dada por:

$$P(Y = c \mid X = x) = \frac{exp(\beta_0 + \sum_i \beta_i x_i)}{1 + exp(\beta_0 + \sum_i \beta_i x_i)}$$

Observa-se que o classificador não é linear. Caso a probabilidade seja maior que 0.5, considera-se a hipótese original - que o evento tem classe c. Caso contrário, descarta-se a hipótese e assume-se que o evento é da outra classe.[11]

2.3.2.Avaliação dos modelos

- Cross-validation

Cross Validation é um procedimento que, ao dividir o dataset usado em diversos subgrupos aleatoriamente escolhidos, possibilita determinar as métricas de um algoritmo de forma não enviesada. Isso ocorre pois os dados de treinamento não são usados para testes, fazendo com que as medidas de eficiência retiradas do classificador não sejam sobre o fato de ele estar acertando sobre aquele dataset específico, mas sim, se ele seria eficiente sobre outros dados [13].

- Overfitting

Overfitting é o nome dado ao fato de treinar um classificador que funciona bem apenas com os dados usados para o treino - ele representa bem aquela amostra, e não o espaço total [13].

- Matriz de confusão

A matriz de confusão é muito útil para visualizar o desempenho de um modelo e facilita o cálculo das métricas seguintes. Trata-se de uma tabela que relaciona as classes reais e as classes atribuídas pelo modelo, mostrando acertos e erros para cada

classe. Para o caso do trabalho, em que há duas classes possíveis - uma verdadeira e outra falsa -, a matriz pode ser montada da seguinte forma [13]:

- Classe real e atribuída verdadeiras: verdadeiro positivo (*True Positive* - TP);
- Classe real verdadeira e atribuída falsa: falso negativo (*False Negative*- FN);
- Classe real falsa e atribuída verdadeira: falso positivo (*False Positive* - FP);
- Classes real e atribuída falsas: verdadeiro negativo (*True Negative* - TN).

	Verdadeiro (modelo)	**Falso (modelo)**
Verdadeiro (real)	TP	FN
Falso (real)	FP	TN

Tabela 7: Matriz de Confusão.

- Acurácia, Precisão e Revocação

A acurácia (*accuracy*) denota a proporção de acertos em relação ao total de observações classificadas. Em termos de TP, TN, FP e FN, a acurácia pode ser escrita como:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

Já a precisão (*precision*) é uma métrica que foca apenas na classe desejada, ou seja, é a relação entre acertos da classe desejada e o seu total de atribuições.

$$Precision = \frac{TP}{TP + FP}$$

Por fim, a revocação (*recall*) também foca na classe desejada como a precisão, mas utiliza a relação entre acertos e o total real da classe.

$$Recall = \frac{TP}{TP + FN}$$

- F_1-score

O F_1-score é outra métrica de desempenho de um modelo. Ele é dado pela média harmônica entre precisão e revocação, sendo muito utilizado em casos que se baseiam em ambas as métricas.

$$F_1 = \frac{2}{precision^{-1} + recall^{-1}} = \frac{TP}{TP + 1/2(FP+FN)}$$

3.METODOLOGIA

Com base nos objetivos, a metodologia deste trabalho pode ser definida em 5 fases: estudo do tema, aquisição da base de dados, extração e análise das características a serem utilizadas, modelagem do sistema de classificação e avaliação dos modelos criados. A fase de estudo do tema está apresentada no capítulo Aspectos Conceituais, enquanto o restante pode ser acessado nesta parte do relatório.

3.1.Base de dados

A base de dados pode ser dividida em duas partes: a primeira é uma base de dados já formada por um outro trabalho, chamada de Fake.br Corpus, enquanto a segunda consiste de dados obtidos durante o projeto, que são as notícias obtidas por *crawler*.

- Fake.br Corpus

O Fake.br Corpus, chamado de Corpus no decorrer do projeto, é uma base de dados contendo notícias verdadeiras e falsas em português brasileiro, como descrito por Monteiro [5]. Essa base consiste de 7200 notícias, sendo 3600 de cada tipo, retiradas dos domínios (*sites*) listados na tabela 8.

Notícias falsas	Notícias verdadeiras
Diário do Brasil	G1
A Folha do Brasil	Folha de São Paulo
The Jornal Brasil	Estadão
Top Five TV	

Tabela 8: Domínios de origem das notícias do Fake.br Corpus.

- Notícias obtidas por crawler

Um dos objetivos secundários é expandir a base de notícias inicial para contemplar assuntos mais atuais, dado que o conteúdo do Corpus datava até o ano de 2018. Para isso, foi utilizada a mesma ferramenta dos criadores do Corpus original: um webcrawler. O grupo optou pelo *news-please*, projeto de código aberto, voltado para extração de notícias de forma automatizada em *sites* fornecidos.

Alguns *sites* são mais facilmente explorados por *webcrawlers*. Assim, utilizou-se esse fator e a reputação para determinar a lista de sites com conteúdo verídico na tabela 9.

g1.globo.com
terra.com.br
oglobo.globo.com
noticias.uol.com.br
cnnbrasil.com

Tabela 9: Domínios utilizados para obter notícias verdadeiras pelo *crawler*

Para determinar os sites com conteúdo potencialmente falso e duvidoso, foi utilizado o Google Fact Check API para realizar um levantamento de fontes que recorrentemente eram desbancadas por entidades de Fact Checking. Como a API exige um termo de pesquisa para retornar os resultados, buscou-se por diversos assuntos atuais que costumam ser envoltos em desinformação, como “Coronavírus”, “Vacina”, “Lula”, “Bolsonaro” e “Eleições”. Alguns sites se destacaram em número de aparições e foram utilizados para alimentar a nova base, como listado na tabela 10:

estudosnacionais.com
jornaldacidadeonline.com.br
imprensabrasil.com.br
conexaopolitica.com.br
terrabrasilnoticias.com
roteirodenoticias.com.br
criticanacional.com.br

Tabela 10: Domínios utilizados para obter notícias verdadeiras pelo *crawler*

- Tratamento da base

Após a obtenção de ambas as bases, foi feito um tratamento que consiste em retirar notícias e partes do texto que não importam, como notícias repetidas ou em

outra língua que não português brasileiro, textos em vazio ou anúncios. Foram utilizados SQL e Python para esta parte do projeto.

3.2. Extração e Análise dos Dados

Antes de realizar a modelagem do sistema, é necessário extrair as características a serem utilizadas no modelo. Também foi feita uma análise dessas características, procurando aquelas que mais influenciam no resultado.

3.2.1.Parâmetros sintáticos

Um dos tipos de características extraídas foram os chamados parâmetros sintáticos. Eles representam valores como quantidade de palavras, pontuações e verbos. No caso, as características extraídas desse tipo seguem na tabela A-1 anexada. Essas *features* foram retiradas de todas as notícias e obtidas por meio da biblioteca *spacy* e *regex* em Python.

A partir dos dados obtidos das características, foram feitas análises para poder selecionar as características mais importantes para o modelo. Inicialmente foi criada a tabela A-2, que serviu para que se tivesse uma ideia sobre as possíveis diferenças entre textos verdadeiros e textos falsos. Nesta tabela, pode-se observar que a grande maioria dos parâmetros não diferem notavelmente. No entanto, ainda foi possível separar algumas *features* mais promissoras para a classificação: *non-immediacy*, *diversity*, *polarity*, *average_sentence_len* e *num_tokens*. Uma análise mais profunda mostrou que *num_tokens* e *diversity* estavam altamente correlacionados, logo o segundo parâmetro foi descartado do grupo. Ainda assim, *non-immediacy*, *polarity*, *num_tokens*, *average_sentence_len* se mostraram fatores diferenciadores. Segue uma análise de cada um deles.

- Número de tokens

Esta métrica representa a quantidade de unidades básicas formadoras do texto, basicamente representa o número de palavras, pontuações e números no texto. Na tabela, vê-se claramente que este parâmetro difere do texto verdadeiro (1075.171386 ± 757.885920) para o texto falso (302.215058 ± 340.553858). Para melhor visualizar esta diferença, fez-se o seguinte histograma na figura 7:

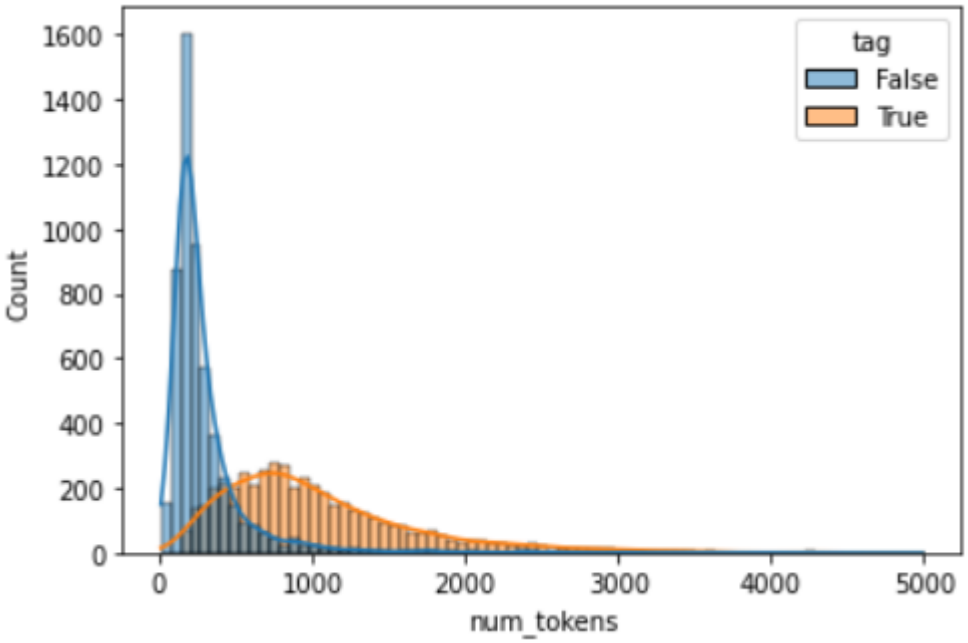

Figura 7: Histograma de número de tokens por classificação

Pelo gráfico (Figura 7), observa-se que as notícias verdadeiras tendem a ser maiores (mais palavras) do que as notícias falsas. Isso pode ser explicado pela característica apelativa da notícia falsa, de forma que notícias menores podem ter maior chance de serem lidas.

- Polarity

Este parâmetro é calculado somando o valor determinado para cada palavra do texto, usando o método de NLP de Análise de Sentimento. O resultado da média deste parâmetro mostrou ser diferente ao comparar o corpus falso (3.121588 ± 5.670435) com o verdadeiro (0.729179 ± 2.516052), um resultado promissor. Pensou-se que talvez esta diferença estivesse relacionada com o fato de textos verdadeiros terem mais palavras do que textos falsos e este parâmetro não ser normalizado em relação ao tamanho do texto. Para identificar uma possível relação, foi realizado o *scatter plot*, conforme figura 8:

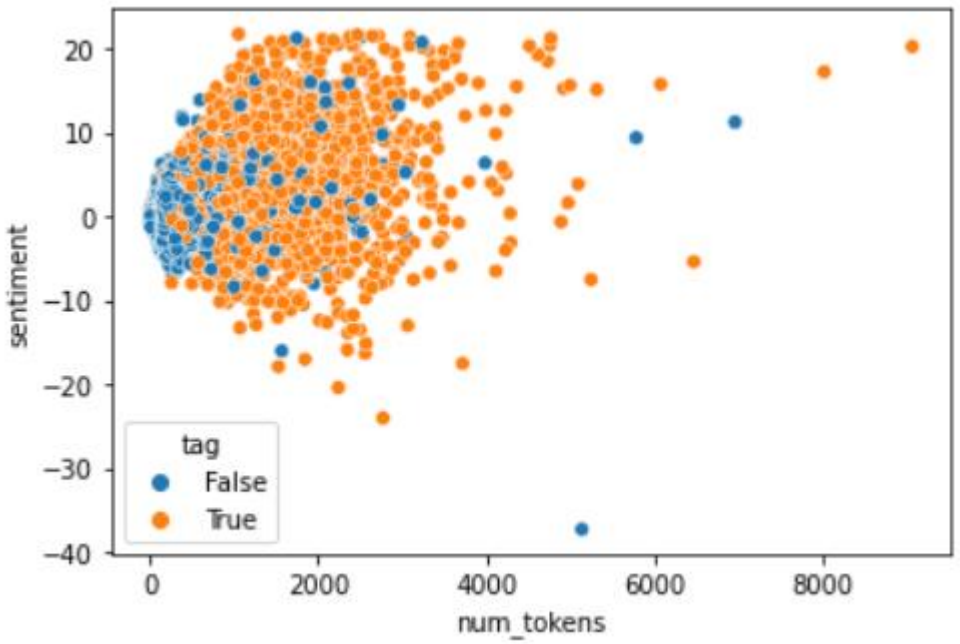

Figura 8: ***Polarity*** **por** ***num_tokens.***

A partir dele, foi possível perceber que não existe correlação entre ambos os parâmetros, mostrando que a diferença nas médias entre os textos verdadeiros e falsos seja significativa. Para confirmar esta hipótese, traçou-se o histograma destes parâmetros.

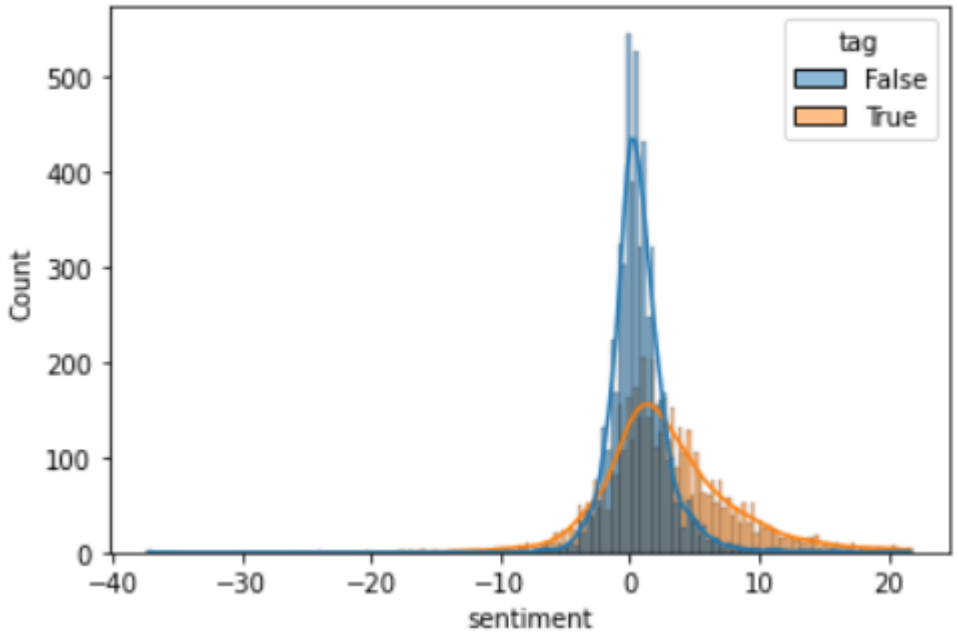

Figura 9: Histograma de polaridade por classificação

O gráfico (Figura 9) mostra que existe uma sobreposição das distribuições, no entanto pode-se ver que textos muito positivos (positividade acima de 5) tendem a ser verdadeiros, em sua maioria, indicando que tal característica poderia ser usada pelo algoritmo de inteligência artificial para diferenciar textos.

Esta diferença pode ser explicada devido ao caráter mais agressivo e negativamente crítico da maioria dos textos falsos, que normalmente tratam de teorias da conspiração, fazem duras críticas a personalidades políticas, entre outros. Ao olhar o universo de textos jornalísticos reais, encontram-se textos negativos, como, por exemplo, notícias de desastres. No entanto, ao mesmo tempo, existem textos com polaridade mais positiva e. g. a notícia sobre os respiradores desenvolvidos pela POLI.

Em suma, poder-se-ia dizer que as *Fake news* buscam imitar as notícias reais, no entanto, quem produz conteúdo falso não está tão interessado em escrever notícias com polaridade positiva. Isto pode estar relacionado ao fato de que notícias com polaridade negativa têm maior chance de serem consideradas verdadeiras com um primeiro olhar [14].

- Non-immediacy

Esta *feature* é calculada somando-se todos os pronomes da primeira e segunda pessoas do plural e do singular presentes no texto. Tais palavras não costumam habitar textos jornalísticos, devido ao fato de estes serem pessoais, no entanto foi possível ver que as estatísticas deste valor variaram quando comparadas as bases falsa (0.620626 ± 1.835849) e verdadeira (3.038462 ± 8.488432). Pensou-se que, por ser um valor não normalizado pelo tamanho do texto, *num_tokens* poderia estar relacionado com essa métrica, assim fez-se o seguinte gráfico (figura 10):

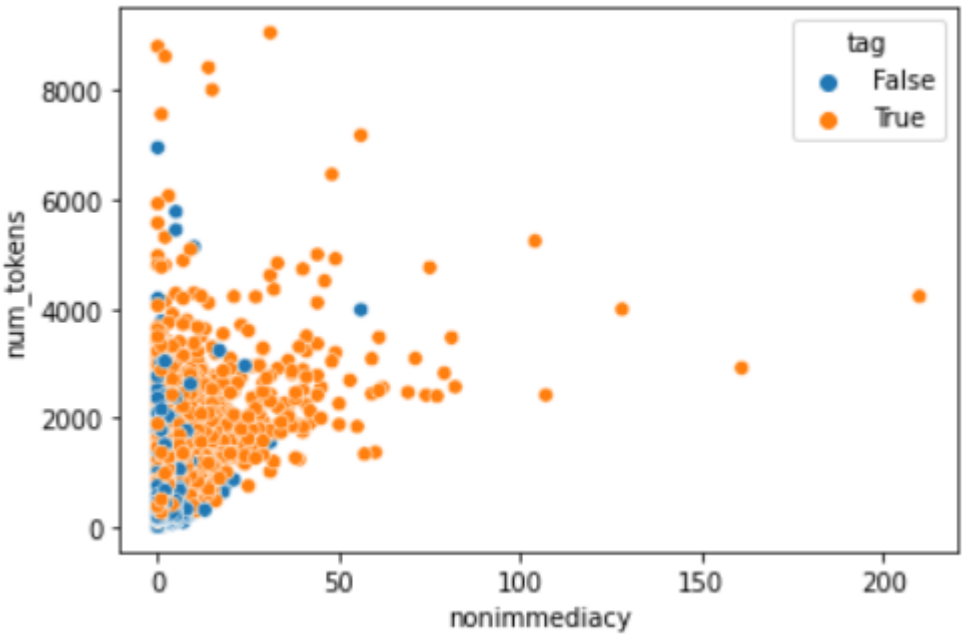

Figura 10: Gráfico de dispersão da iminência

O gráfico mostra uma correlação muito baixa entre os valores, indicando que possivelmente a *feature* em questão poderia ser uma diferenciadora entre os textos. Para melhor visualizar os dados, considerando que muitos ficavam concentrados em zero e também bem espalhados, fez-se o seguinte gráfico (figura 11):

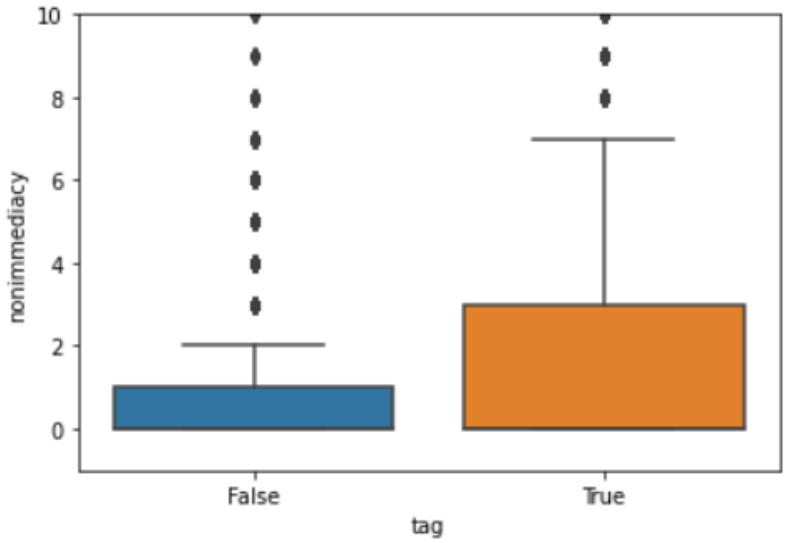

Figura 11: Gráfico em blocos de *non-immediacy*.

A partir do gráfico (figura 11), é possível observar que os valores de *non-immediacy* tendem a ser maiores nas notícias verdadeiras, ou seja, há um maior uso de

pronomes que referenciam o próprio locutor ou escritor. Após uma pequena inspeção dos textos com grande *non-immediacy*, os casos foram atribuídos a respostas de entrevistas.

- Average sentence length

Este parâmetro representa o tamanho médio de palavras nas sentenças em um texto. A partir da tabela A-1. pode-se ter uma ideia de que ele é maior para as notícias verdadeiras. Desta forma, decidiu-se fazer o histograma da figura 12.

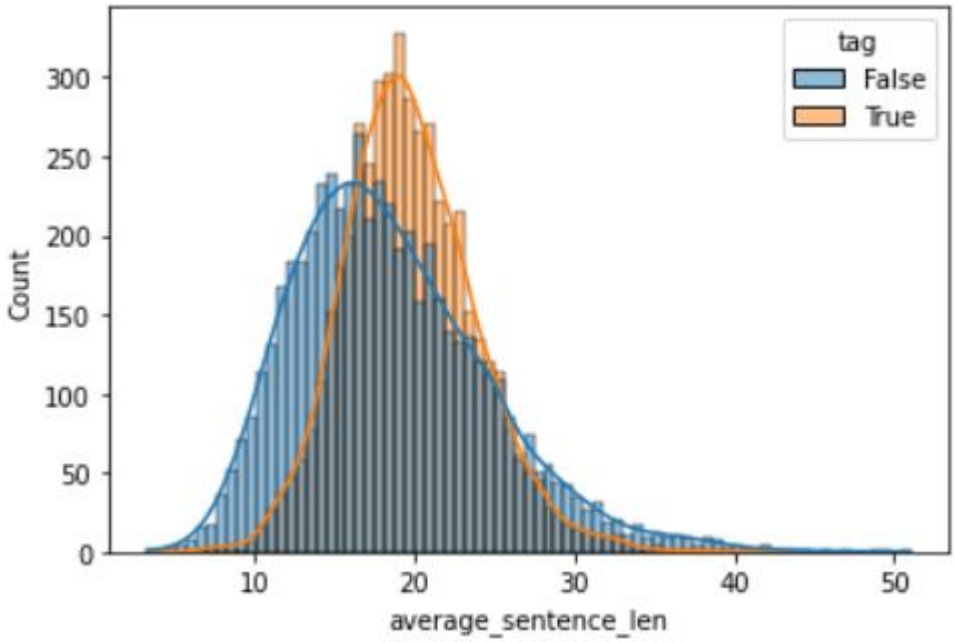

Figura 12: Histograma do parâmetro *average_sentence_len*.

Tal gráfico (figura 12) é interessante por mostrar que os textos verdadeiros possuem uma estimativa de média maior para o parâmetro, o que pode indicar, no geral, uma capacidade inferior para compor orações subordinadas por parte dos escritores de fake news. Ao mesmo tempo, a variância da curva para as notícias verdadeiras é menor, o que indica que estas tendem a seguir um padrão de escrita não observado nos textos falsos. Por fim, a alta concentração de textos falsos com valores baixos para o parâmetro (menor do que 15) e também a maior concentração destes textos na região acima de 30 mostram duas features que poderiam ser usadas pelo algoritmo preditor para diferenciar notícias.

3.2.2.Parâmetros semânticos (Bag of words)

Outro tipo de característica extraída foi o *bag of words*, chamado de parâmetro semântico, por se basear nas próprias palavras do texto. Um estudo preliminar por WordCloud, figura 13, mostra potencial no uso desses parâmetros.

Figura 13: CloudWords das notícias.

Observa-se na figura 13 que há várias palavras repetidas entre ambas, mas a frequência (tamanho da palavra) é diferente. Nota-se que a representação dos termos mais comuns em notícias falsas também apresenta mais palavras relacionadas com temas normalmente propensos à desinformação, geralmente na área de política, medidas públicas e saúde. Assim, é possível realizar algumas observações:

- Nomes de políticos como “Dilma”, “Lula” e “Bolsonaro” aparecem com maior frequência na base de notícias falsas, bem como a sigla do partido “PT”.
- Termos como “esquerda” e “petista” também são mais proeminentes na base falsa.
- “Ex presidente” é uma sequência de palavras comum na base falsa que não é tão comum na verdadeira.
- No tema saúde pública, na base verdadeira observa-se que são recorrentes os tópicos “COVID” e “Vacina”. Na base falsa a palavra “Aborto” é comum.
- Na base verdadeira, “Direito” e “Justiça” são recorrentes, enquanto na base falsa, “STF” e “Sérgio Moro” são mais presentes. “Lava-Jato” é um tópico comum entre elas.

Nota-se que existe significativa diferença nos temas e termos dentre as diferentes classificações das fontes. Embora a política seja um tópico muito comum em ambas, o tratamento é diferente e mais imparcial na base *fake*. A presença comum da palavra “aborto”, por exemplo, na base falsa, associa-se com um assunto usual entre classes mais conservadoras, sendo esse grupo frequentemente associado com a difusão de desinformação [15]. Esse comportamento era esperado e demonstra que a utilização do *bag of words*, através do qual essas frequências serão quantificadas e consideradas na classificação, pode trazer bons resultados.

3.3.Modelagem

Para aplicar os modelos, utilizou-se a linguagem Python pois dispõe de bibliotecas especializadas como scikit-learn, com funções próprias para a construção e treinamento de modelos, pandas, para manipulação de dados em formato de tabela, e Flask para a construir as requisições web (APIs). Assim, foram montados os códigos iniciais para o teste do desempenho dos modelos: KNN, SVC, Random Forest e Perceptron, para os parâmetros sintáticos, e dos modelos KNN, SVC, Random Forest, Perceptron, Naive Bayes e Logistic Regression. Todos os programas montados para o trabalho, bem como as suas documentações, estão disponíveis no repositório em https://github.com/ozazas/alethea.

O código consiste essencialmente em três partes: pré-processamento dos dados da base, montagem do modelo e avaliação dos mesmos por *cross-validation*.

3.3.1. Pré-processamento

- Parâmetros sintáticos

Como pré-processamento dos chamados parâmetros sintáticos, estes foram divididos em três grupos, chamados de *pools*, como na tabela 11. Isso foi feito para observar os efeitos de cada conjunto na classificação e quais parâmetros são mais pertinentes. Por exemplo, o *pool* 3 consiste das características mais promissoras determinadas pela análise dos dados. Em nenhum deles foi utilizado a característica *num_tokens*, pois não é desejado que o modelo esteja enviesado em relação ao tamanho da notícia.

pool 1	pool 2	pool 3
tag	tag	tag
pausality	pausality	pausality
average_sentence_len	average_sentence_len	average_sentence_len
nonimmediacy	nonimmediacy	nonimmediacy
sa_pos_score	sa_pos_score	sa_pos_score
sa_neg_score	sa_neg_score	sa_neg_score
average_word_len	average_word_len	
percentage_speeling_errors	percentage_speeling_errors	
emotiveness	emotiveness	
diversity	diversity	
average_quote_len	average_quote_len	
`+ 30 colunas (numéricas)		

Tabela 11: Dados contidos em cada pool utilizado.

Então foram criados três arquivos csv para cada pool: um para toda a base, um só com a base do corpus e um só com a do crawler, para poder avaliar comparativamente a influência das bases, bem como o comportamento decorrente dos parâmetros utilizados.

Ao carregar o arquivo csv, o programa normaliza as colunas numéricas pelo número de tokens e separa a classificação, coluna tag, dos parâmetros. Em seguida. foi separada aleatoriamente a base em 5 grupos de teste para realizar o cross-validation.

- Parâmetros semânticos (*bag of words*)

O pré-processamento dos textos para o uso do *bag of words* consiste em extrair as palavras, transformá-las todas para caixa baixa, retirar as *stopwords* e, por fim, aplicar um *stemmer*. Das palavras que restaram, foram contadas no *bag of words* apenas aquelas que aparecem minimamente uma certa quantidade de vezes em todos os textos de treinamento, a fim de retirar palavras pouco frequentes, como palavras estrangeiras e com erros ortográficos.

3.3.2. Montagem do modelo

Para a montagem do modelo, utilizou-se a biblioteca *scikit-learn*, também conhecida como *sklearn*, e para cada um dos modelos foram utilizados os seguintes parâmetros[16][17]:

➔ KNN:
clf = neighbors.KNeighborsClassifier(n_neighbors=j, weights=uniform)
com j variando entre 11 e 51 com passo 2.

➔ SVC
o classificador SVC não possui parâmetros, clf = svm.SVC()

➔ Random Forest
clf = sk.ensemble.RandomForestClassifier(n_estimators=200, criterion='gini', max_depth=j)
com j entre 5 e 21 com passo 2.

➔ Perceptron
clf = neural_network.MLPClassifier(hidden_layer_sizes=100, random_state=0, max_iter=1000)

➔ Naive Bayes
clf = GaussianNB()

➔ Logistic Regression
clf = sk.linear_model.LogisticRegression()

3.3.3. Avaliação dos modelos

Para avaliar os classificadores e os efeitos dos diferentes pools utilizados para cada um dos grupos de teste, treinou-se o modelo com os grupos restantes e foram obtidos os valores de acurácia, precisão e recall, comparado os valores obtidos pelo modelo com as respostas do grupo de teste e, ao final, foi tirada uma média desses valores. Para o caso de KNN, testou-se para diferentes números de vizinhos e foi escolhido o melhor resultado, e, para o caso de Árvore de Decisão, também foi escolhida a profundidade que fornecia os melhores resultados. Nesses casos, para escolher o melhor ponto, procurou-se o valor que fornecesse a melhor acurácia, recall, precisão e f-score, como no exemplo a seguir (figura 14):

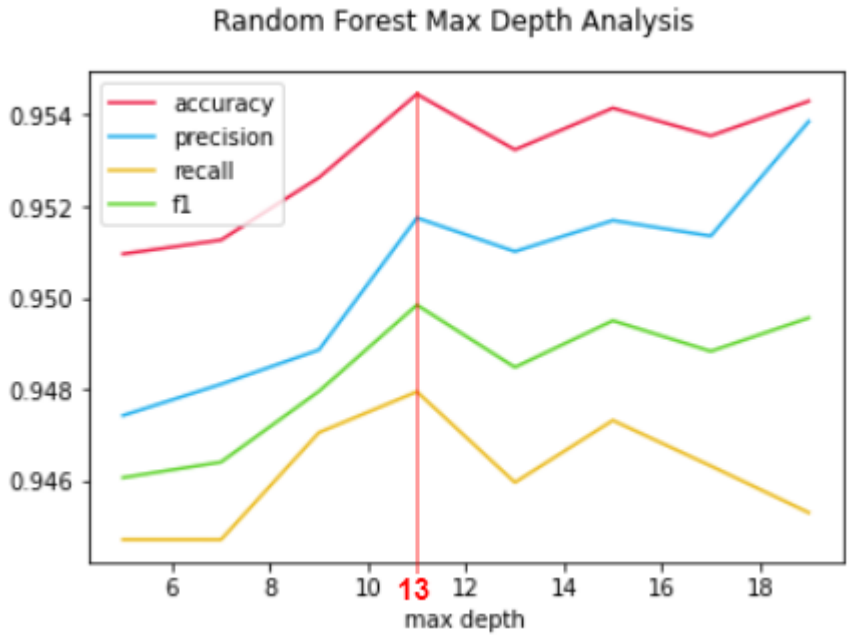

Figura 14: Métricas para diferentes profundidades de Árvore de Decisão.

3.4.Construção do serviço web

O serviço web - que não é o foco deste trabalho - nasceu como uma maneira de permitir que o poder do classificador estivesse ao alcance de qualquer um com acesso à internet, sem que fossem necessários maiores conhecimentos em programação.

Assim, buscou-se criar algo que fosse funcional, sem deixar de lado a simplicidade. Para tanto, buscou-se um template HTML5 que fosse facilmente modificável [18], e, para a parte de subir o código para a internet, considerando que este precisa de uma certa quantidade de processamento e memória, dada a natureza do classificador, seguiu-se o tutorial do AWS lightsail [19].

- **Desenvolvimento das ferramentas de requisição**

A ferramenta de requisição consistiu em, basicamente, fazer um compilado dos códigos que tratam os textos - transformando-os em features - e acoplando a saída desses códigos a classificadores pré-treinados. Por fim, expôs-se esse fluxo de decompor os textos em features e classificá-los através de um endpoint de uma API escrita em flask [20].

- **Construção do servidor**

O servidor, em poucas palavras, é uma máquina na qual dentro está rodando a API desenvolvida. O processo de subir APIs em servidores, por muito tempo, foi complexo e demorado. Felizmente, hoje em dia há os serviços *cloud*, que permitem reservar um espaço da internet para rodar código facilmente [22]. Assim, foi transformado o código da API em conjunto do código HTML em um container Docker, e, facilmente, foi enviado para a *cloud* da AWS. por meio da interface *Lightsail.*

- **Desenvolvimento do Frontend**

O *Frontend é* a interface gráfica que permite ao usuário classificar as notícias, ou seja, é o *site* que carrega no navegador. O processo de concepção foi pautado por utilizar um template HTML5 e fazer pequenas modificações. Principalmente, as modificações consistiram em adicionar um pouco de *javascript* para permitir que, por meio de caixas de textos e botões, o usuário interagisse com a API. Ademais, algumas mudanças foram feitas no CSS, para mostrar a tela como verde quando o classificador considerava o texto como provavelmente verdadeiro, e vermelha no contrário.

4.RESULTADOS

4.1.Base de dados

Inicialmente foram obtidas pelo crawler 8828 notícias, sendo 3593 de domínios confiáveis e 5235 dos *sites* considerados suspeitos. A tabela 12 apresenta as quantidades de notícias de cada domínio.

Site	Links
g1.globo.com	663
www.terra.com.br	704
oglobo.globo.com	1395
noticias.uol.com.br	54
cnnbrasil.com	777
Total	3593

Site	Links
estudosnacionais.com	783
jornaldacidadeonline.com.br	26
imprensabrasil.com.br	8
conexaopolitica.com.br	142
terrabrasilnoticias.com	11
roteirodenoticias.com.br	3080
criticanacional.com.br	1185
Total	5235

Tabela 12: quantidades de notícias retiradas por site, notícias verdadeiras na tabela da esquerda e notícias falsas na tabela da direita, para a construção da base

Após a junção com o Corpus e a filtragem dos dados, foi formada a base chamada de *full*. As quantidades finais podem ser observadas na tabela 13.

Notícia	Fake.br Corpus	Obtidas por *crawler*	Total (*full*)
Verdadeira	2997	1713	4710
Falsa	3589	1869	5458
Total	6586	3582	10168

Tabela 13: quantidade de notícias verdadeiras e falsas em cada base de dados

4.2.Análise dos Modelos

- Parâmetros Sintáticos

Aplicando o método para escolha dos melhores parâmetros para todas as combinações de pools, foram obtidos os seguintes resultados (tabela 14) com os valores melhores valores de k:

	KNN (número de vizinhos)		
	crawler	**corpus**	**full**
pool 1	31	47	37
pool 2	33	45	17
pool 3	33	37	42

	Random Forest (profundidade)		
	crawler	**corpus**	**full**
pool 1	15	11	13
pool 2	15	13	17
pool 3	9	13	9

Tabela 14: Melhores valores de k para diferentes combinações de pool e base para os modelos de KNN e Random Forest.

Utilizando esses valores nos parâmetros dos modelos, foram obtidas as tabelas do Anexo B com os valores de acurácia, precisão, *recall* e *F-score* para as diferentes combinações de pool e modelo e base, tendo destacado em vermelho os melhores valores de precisão e em verde os piores, sendo os valores em células coloridas com maior saturação valores maiores.

Assim, tem-se que as combinações de modelo fonte e pool que apresentaram os melhores resultados, foram (Tabela 15):

Modelo	**Pool**	**Base**	**Precisão**	**Acurácia**
Random Forest	Pool 1	Corpus	95,17%	95,44%
Perceptron	Pool 1	Corpus	94,78%	95,14%
Random Forest	Pool 2	Corpus	94,06%	94,64%

Tabela 15: Melhores resultados obtidos para acurácia e precisão dos modelos criados anteriormente após refino.

O modelo Random Forest mostrou bons resultados. sendo o escolhido para ser aplicado na análise dos parâmetros sintáticos. A análise também demonstrou que a base do Corpus apresenta melhores resultados e que o maior número de parâmetros do Pool 1 também contribuíram para a melhora dos resultados.

Em nota, a base de dados obtida pelo *crawler* também teve melhor desempenho com os modelos Random Forest e Perceptron, porém, ao contrário da base do corpus, que em todos os modelos apresentou melhor desempenho com o maior número de

parâmetros, Pool 1, o crawler nos modelos KNN e SVC apresentaram melhores resultados, com menos parâmetros, Pool 2 e Pool3.

- Parâmetros semânticos

Como resultados dos parâmetros semânticos, houve um total de 43308 diferentes palavras pré-processadas, incluindo palavras chinesas e árabes que acabaram passando pela filtragem inicial. Assim, para retirar esses casos indesejados, as palavras utilizadas nesse método foram limitadas àquelas que aparecem pelo menos 200 vezes no *dataset* de treino e ocorrem em ambas as notícias falsas e verdadeiras. Dessa forma, os melhores resultados dos modelos avaliados podem ser observados na tabela 16.

Modelo	Base	Acurácia	Precisão
KNN	Corpus	60,04%	99,39%
Naive Bayes	Corpus	93,46%	92,62%
Random Forest	Corpus	95,96%	96,85%
Logistic Regression	Corpus	95,90%	97,90%
SVC	Corpus	96,13%	97,11%
Perceptron	Corpus	96,34%	96,81%

Tabela 16: Melhores resultados para cada tipo de modelo para *bag of words*.

Dos resultados obtidos, observa-se que a maioria é ligeiramente melhor que os modelos anteriores, sendo o com melhor acurácia o Perceptron e com melhor precisão a regressão logística. O KNN obteve a menor acurácia, comparado com os outros casos, mas a maior precisão. Nota-se também que, em todos os casos, a melhor base foi a do Corpus novamente. A base do *crawler* também apresentou bons resultados, em torno de 90% de acurácia e precisão, mas não conseguiu resultados melhores que a base do Corpus.

Durante testes feitos manualmente, entretanto, os modelos mostraram viés para classificação de notícias pequenas como falsas e grandes como verdadeiras. Para mitigar esse problema, foi utilizada uma forma alternativa do *bag of words*, em que se considera apenas se há incidência da palavra no texto, não contando a quantidade de

ocorrências. Repetindo o mesmo processo, foram obtidos os resultados descritos na tabela 17.

Modelo	Base	Acurácia	Precisão
KNN	Corpus	62,09%	99,80%
Naive Bayes	Corpus	93,53%	91,99%
Random Forest	Corpus	94,85%	96,79%
Logistic Regression	Corpus	96,10%	95,94%
SVC	Corpus	95,29%	94,95%
Perceptron	Corpus	95,56%	95,50%

Tabela 17: Melhores resultados para cada tipo de modelo para *bag of words* alternativo.

Os resultados são bem parecidos com os obtidos anteriormente, porém o modelo escolhido dessa vez foi o Logistic Regression, com acurácia e precisão ainda acima de 95%. Em testes manuais com outros usuários, observou-se menor viés, mas ainda existente.

4.3.Construção do serviço

A página da web com os resultados do projeto foi feita com sucesso, e pode ser acessada inicialmente em: <https://alethea-web-service.qeh9262ium3jq.us-east-2.cs.amazonlightsail.com/index.html>.

5.DISCUSSÃO

Ao final do trabalho, buscando validar os resultados, foram feitos testes isolados que permitiram quantificar a acurácia e a precisão do algoritmo e também testes de aceitação do usuário, os quais consistiram em passar o link do site para o usuário e pedir para que ele o testasse com notícias.

Os testes isolados apresentaram métricas que indicam que o modelo funciona bem, com valores acima de 95% para precisão e acurácia. No entanto, na prática, pode ser que eles não tenham sido suficientes, considerando que existe um componente temporal que pode inviabilizar o classificador com o tempo.

Agora, tratando dos testes com usuários, foi apontado que o classificador parecia estar enviesado pelo tamanho do texto, de forma que textos suficientemente grandes sempre seriam considerados verdadeiros e textos suficientemente pequenos sempre seriam considerados falsos. Feitos mais alguns testes, pode-se confirmar isso na prática. Uma possível explicação é o fato de que, em maioria, as notícias verdadeiras são maiores do que as falsas, e, mesmo retirando este parâmetro na hora de treinar o classificador, ele influencia a outros. Um exemplo é a normalização, a qual, apesar de parecer retirar o viés do tamanho do texto, pode, em casos extremos, zerar parâmetros que, apesar de serem sintáticos, não têm uma relação com o tamanho do texto. Uma forma de melhorar isso seria decidir normalizar ou não um dado parâmetro sintático, baseado numa análise prévia de correlação com o tamanho do texto. Importante notar que, para treinar o classificador, não foi passado explicitamente o parâmetro que diz o número de palavras no texto.

Sendo assim, foi possível entregar um protótipo de site que permite a usuários classificarem textos. No entanto, os resultados - principalmente os testes com usuários - mostraram que o classificador precisa ser melhor ajustado para que o aplicativo possa ser amplamente utilizado.

6.CONSIDERAÇÕES FINAIS

6.1.Conclusões do Projeto de Formatura

Ao final deste trabalho, foi desenvolvida uma ferramenta para a identificação de textos com estrutura de textos falsos. Por melhor que sejam os resultados, esse trabalho possui caráter educacional e, portanto, não foi testado exaustivamente para ser um método confiável de identificação de notícias falsas, os resultados apresentam apenas parâmetros similares aos encontrados em notícias falsas. Além do aprendizado de máquina ser fundamentado em probabilidade, o resultado é muito baseado na estrutura dos textos da base, que podem não contemplar muitas variedades de notícias. Porém, ele ainda pode ser utilizado para identificar textos mal-escritos, como foi hipotetizado na análise dos dados e que, muitas vezes, é uma característica de *fake news*, ou que possuem a estrutura das *fake news* da base.

6.2.Trabalhos Futuros

Por conta do tempo limitado, não foi possível abordar, nesta etapa, medidas para tornar as bases de dados mais robustas e/ou desenvolver a base do crawler, para que tenha um desempenho similar ou melhor que o do corpus, nem foi possível aplicar métodos e avaliar mudanças que poderiam beneficiar o desempenho dos modelos estudados.

Porém, algumas ações que podem ser tomadas em trabalhos futuros, nesse sentido de tornar as bases mais robustas e melhorar a precisão dos modelos que foram encontrados ao longo desta pesquisa, são:

- Aplicação de Boost como o XGBoost [23] e AdaBoost [24] para melhorar a performance dos algoritmos de aprendizado de máquina.
- Refino do dataset, eliminando linhas repetidas ou com dados faltantes.

Algumas outras abordagens que podem ser exploradas são:

- Aplicação de outros métodos de classificação.
- Utilização de outras formas de representação de texto como o TF-IDF [25].
- Aplicação de peso, weight, em determinados parâmetros de entrada do classificador.

LISTA DE REFERÊNCIAS

[1] CARVALHO, M. F. C. de; MATEUS, C. A. **Fake News e Desinformação no Meio Digital: Análise da Produção Científica Sobre o Tema na Área de Ciência da Informação**. Múltiplos Olhares em Ciência da Informação, [S.l.], v.8, n.2, 2018. Disponível em: <https://periodicos.ufmg.br/index.php/moci/article/view/16901>. Acesso em: 05 maio 2021.

[2] WARDLE, C. **Fake news: It's complicated**. First Draft, 2017. Disponível em: <https://firstdraftnews.org/latest/fake-news-complicated/>. Acesso em: 05 maio 2021.

[3] PAPANASTASIOU, Y. **Fake News Propagation and Detection**: A Sequential Mode. 2018. Disponível em: <https://yiangos-papanastasiou.com/fake_news_2018.pdf>. Acesso em: 05 maio 2021.

[4] LIM, C. CHECKING HOW FACT-CHECKERS CHECK. **Research & Politics**. Julho de 2018. doi:10.1177/2053168018786848

[5] MONTEIRO R.A., SANTOS R.L.S., PARDO T.A.S., de ALMEIDA T.A., RUIZ E.E.S., Vale O.A. (2018) Contributions to the Study of Fake News in Portuguese: New Corpus and Automatic Detection Results. In: Villavicencio A. et al. (eds) **Computational Processing of the Portuguese Language**. PROPOR 2018. Lecture Notes in Computer Science, vol 11122. Springer, Cham

[6] FREITAS, C. Sobre a Construção de um Léxico da Afetividade para o Processamento Computacional do Português. **Revista Brasileira de Linguística Aplicada** [online]. 2013, v. 13, n. 4, pp. 1031-1059. Disponível em: <https://doi.org/10.1590/S1984-63982013005000024>. Acesso em 11 jul. 2021

[7] HARRISON, H, **Machine Learning Basics with the K-Nearest Neighbors Algorithm,** Towards Data Science, 2018. Disponível em: <https://towardsdatascience.com/machine-learning-basics-with-the-k-nearest-neighbors-algorithm-6a6e71d01761>. Acesso em set. 2021.

[8] GANDHI, R**. Support Vector Machine — Introduction to Machine Learning Algorithms.** Towards Data Science, 2018. Disponível em: <https://towardsdatascience.com/support-vector-machine-introduction-to-machine-learning-algorithms-934a444fca47>. Acesso em set. 2021.

[9] GUPTA, P. **Decision Trees in Machine Learning.** Towards Data Science, 2017. Disponível em: <https://towardsdatascience.com/decision-trees-in-machine-learning-641b9c4e8052>. Acesso em set. 2021.

[10] ZHOU, V. **Machine Learning for Beginners: An Introduction to Neural Networks.** Towards Data Science, 2019. Disponível em: <https://towardsdatascience.com/machine-learning-for-beginners-an-introduction-to-neural-networks-d49f22d238f9 >. Acesso em set. 2021.

[11] PANT, A. **Introduction to Logistic Regression.** Towards Data Science, 2019. Disponível em: <https://towardsdatascience.com/introduction-to-logistic-regression-66248243c148>. Acesso em set. 2021.

[12] **Supervised Learning**. Disponível em: <https://scikit-learn.org/stable/supervised_learning.html#supervised-learning >. Acesso em set 2021.

[13] JAMES, G.; HASTIE, T.; TIBSHIRANI, R.; WITTEN, D. **An Introduction to Statistical Learning**. Springer. 2013.

[14] JAFFÉ, M. E.; GREIFENEDER, R. **Can that be true or is it just fake news? New perspectives on the negativity bias in judgments of truth**. The Psychology of Fake News. 2021. v.1, p.115-130. Disponível em: <https://library.oapen.org/bitstream/handle/20.500.12657/46921/9781000179033.pdf?sequence=1>. Acesso em 17 jul. 2021

[15] AZEVEDO, M. De C.; LIMA, M. A. A. (2020). **Fake News e Pós-Verdade na Construção do Neoconservadorismo no Brasil Pós-2013 e os Efeitos nas Eleições de 2018.** Letrônica, 13(2), E35546. Disponível em <https://doi.org/10.15448/1984-4301.2020.2.35546>. Acesso em 11 jul. 2021

[16] **sklearn.feature_extraction.text.CountVectorizer**. Documentação da função CountVectorizer. Disponível em: <https://scikit-learn.org/stable/modules/generated/sklearn.feature_extraction.text.CountVectorizer.html>. Acesso em: 06 nov. 2021.

[17] **sklearn.naive_bayes.GaussianNB**. Documentação Naive Bayes. Disponível em: <https://scikit-learn.org/stable/modules/generated/sklearn.naive_bayes.GaussianNB.html>. Acesso em: 06 nov. 2021.

[17] **AWS Lightsail**. Documentação do AWS Lightsail. Disponível em: <https://docs.aws.amazon.com/lightsail/ >. Acesso em: 18 dez. 2021.

[18] **HTML5 UP!**. Acesso a exemplos de templates de sites. Acesso em: <https://html5up.net/>. Acesso em:06 nov. 2021.

[19] **AWS Lightsail flask RESTful API walkthrough**. Guia sobre servir uma aplicação *flask* . Disponível em: <https://aws.amazon.com/pt/getting-started/hands-on/serve-a-flask-app/ >. Acesso em: 18 dez. 2021.

[20] **Flask-RESTX**. Documentação do Flask-RESTX. Disponível em: <https://flask-restx.readthedocs.io/en/latest/>. Acesso em: 06 nov. 2021.

[21] **Definição de Infrastructure as a Service pela microsoft.** Disponível em: <https://azure.microsoft.com/pt-br/overview/what-is-iaas/>. Acesso em: 18 dez. 2021.

[22] **Definição de cloud pela microsof**t. Disponível em: <https://azure.microsoft.com/en-us/overview/what-is-the-cloud/>. Acesso em: 18 jan. 2022

[23] **The turing test.** Disponível em: <https://plato.stanford.edu/entries/turing-test/>. Acesso em: 30/01/2022

[23] **XGBoost**. Documentação do XGBoost. Disponível em: <https://xgboost.readthedocs.io/en/stable/>. Acesso em: 18 dez. 2021.

[24] **AdaBoost Classifier in Python**. Datacamp, 2018. Disponível em: <https://www.datacamp.com/community/tutorials/adaboost-classifier-python>. Acesso em: 18 dez. 2021.

[25] **TF-IDF/Term Frequency Technique: Easiest explanation for Text classification in NLP using Python (Chatbot training on words).** Disponível em: <https://medium.com/analytics-vidhya/tf-idf-term-frequency-technique-easiest-explanation-for-text-classification-in-nlp-with-code-8ca3912e58c3>. Acesso em: 18 dez. 2021.

[26] OKANO, E.Y. **Análise e caracterização de textos intencionalmente enganosos escritos em português usando métodos de processamento de textos**. Monografia (Mestrado em Computação Aplicada) - Faculdade de Filosofia, Ciências e Letras de Ribeirão Preto, Universidade de São Paulo. Ribeirão Preto - SP, p. 109. 2020.

ANEXO A

Nome	**Definição**
num_tokens	Nº de *tokens* no texto
num_word_without_punct	Nº de palavras total no texto
num_types	Nº de palavras diferentes no texto
num_links	Nº de *links*
num_words_upper	Nº de palavras totalmente em caixa alta
num_verbs	Nº de verbos total
num_subj_imp_verbs	Nº de verbos no subjuntivo e imperativo
num_nouns	Nº de substantivos total
num_adjectives	Nº de adjetivos
num_adverbs	Nº de advérbios
num_modal_verb	Nº de verbos auxiliares
num_singular_pronouns	Nº de pronomes em primeira pessoa do singular
num_plural_pronouns	Nº de pronomes em primeira pessoa do plural e pronomes
num_pronouns	Nº de pronomes total
pausality	*Pausality*
num_characters	Nº de caracteres totais no texto (sem contar espaços e afins)
average_sentence_len	Média de palavras por frase
average_word_len	Média de caracteres por palavra
percentage_speeling_errors	Nº de palavras com erro ortográfico por nº de palavras totais
emotiveness	*Emotiveness*
diversity	*Diversity*
num_ponctuation	Nº de pontuações
num_exclamation	Nº de pontos de exclamação
num_question_mark	Nº de pontos de interrogação
num_quotes	Nº de citações e falas (entre aspas)

average_quote_len	Média de palavras nas citações e falas
num_cconj	Nº de CCONJ (*POS-tag*)
num_sconj	Nº de SCONJ (*POS-tag*)
num_interjections	Nº de interjeições
num_numbers	Nº de números
num_dets	Nº de DETS (*POS-tag*)
num_verb_conditional	Nº de verbos condicionais
num_verb_gerund	Nº de verbos no gerúndio
num_verb_infinitive	Nº de verbos no infinitivo
num_verb_participle	Nº de verbos no particípio
num_verb_indicative	Nº de verbos no indicativo
nonimmediacy	*Non-immediacy*
sa_pos_score	Grau positivo da análise de sentimento dado ao texto
sa_neg_score	Grau negativo da análise de sentimento dado ao texto
(Característica)_norm	Determinada característica normalizada por num_tokens

Tabela A-1: Características extraídas dos textos e definições.

	Verdadeiro		Falso	
métrica	μ	σ	μ	σ
num_tokens	1075.171386	757.885920	302.215058	340.553858
pausality	3.046718	0.690935	2.956950	0.982400
average_sentence_len	19.756033	4.130360	18.521599	6.170744
average_word_len	4.876428	0.217808	4.902777	0.297733
percentage_speeling_errors	0.013240	0.006979	0.015035	0.011789
emotiveness	0.188851	0.056596	0.192524	0.077931
diversity	0.493296	0.078971	0.656528	0.088690
average_quote_len	21.548603	41.887098	12.271045	19.616849
nonimmediacy	3.038462	8.488432	0.620626	1.835849
polarity	3.121588	5.670435	0.729179	2.516052
diversity_truncated	0.640451	0.052392	0.686444	0.062693
num_word_without_punct_norm	0.872795	0.024577	0.865779	0.038794
num_types_norm	0.430663	0.070661	0.567424	0.073450
num_words_upper_norm	0.011946	0.010829	0.013902	0.020878
num_verbs_norm	0.130419	0.023928	0.137180	0.031248
num_subj_imp_verbs_norm	0.004406	0.003315	0.004848	0.006117
num_nouns_norm	0.286126	0.035637	0.285459	0.045184
num_adjectives_norm	0.043783	0.015080	0.045415	0.021139
num_adverbs_norm	0.033841	0.011926	0.034327	0.017226
num_modal_verb_norm	0.015152	0.006870	0.016752	0.011040
num_singular_pronouns_norm	0.001689	0.003878	0.001664	0.004933
num_plural_pronouns_norm	0.000533	0.001287	0.000491	0.002042
num_pronouns_norm	0.037766	0.018088	0.039778	0.022564
num_characters_norm	4.256504	0.230943	4.247056	0.348232
num_ponctuation_norm	0.135972	0.022533	0.142978	0.036138
num_exclamation_norm	0.000251	0.000975	0.002432	0.006104
num_question_mark_norm	0.000971	0.002434	0.001865	0.005692
num_quotes_norm	0.007363	0.005878	0.011537	0.010528
num_cconj_norm	0.024704	0.007729	0.023858	0.011402
num_sconj_norm	0.026562	0.010830	0.026323	0.014577
num_interjections_norm	0.000154	0.000572	0.000355	0.001712
num_numbers_norm	0.022378	0.016646	0.017938	0.019840
num_dets_norm	0.081488	0.013783	0.082396	0.020617
num_verb_conditional_norm	0.001936	0.002321	0.002390	0.004350
num_verb_gerund_norm	0.004546	0.003453	0.006460	0.007044
num_verb_infinitive_norm	0.023567	0.010133	0.023563	0.014082
num_verb_participle_norm	0.020389	0.008592	0.020878	0.012466
num_verb_indicative_norm	0.075537	0.016366	0.078898	0.023151
num_bad_words_norm	0.000059	0.000403	0.000225	0.001418
nonimmediacy_norm	0.002221	0.004312	0.002156	0.005423
ent_misc	0.009216	0.006227	0.010507	0.008831
ent_loc	0.020978	0.011787	0.019101	0.014370
ent_per	0.019622	0.013334	0.026286	0.019098
ent_org	0.011298	0.007942	0.014350	0.010567

Tabela A-2: Médias e desvios padrões das características obtidas.

ANEXO B

	Pool 1		
	SVC - crawler	SVC - corpus	SVC - full
Acurácia	0,7152505396	0,9357733526	0,8575914326
Precisão	0,6939898365	0,9443968148	0,846137813
Recall	0,723913764	0,912583334	0,8464120695
F-Score	0,7195560771	0,9240328689	0,8519650791
	Pool 2		
	SVC - crawler	SVC - corpus	SVC - full
Acurácia	0,7194326921	0,9069246218	0,8288740211
Precisão	0,7414759851	0,9164086011	0,8343677651
Recall	0,6352893025	0,8751768249	0,786717864
F-Score	0,6747478744	0,8907679328	0,8072459455
	Pool 3		
	SVC - crawler	SVC - corpus	SVC - full
Acurácia	0,7238980700	0,9047982321	0,82916949
Precisão	0,7320972464	0,9150681132	0,8267372293
Recall	0,6668276344	0,8715212523	0,7986771426
F-Score	0,6941918684	0,8878480422	0,8136377295

	Pool 1		
	KNN - crawler	KNN - corpus	KNN - full
Acurácia	0,7107804867	0,940025095	0,8573957915
Precisão	0,6792906866	0,9386577296	0,8385476581
Recall	0,7483723263	0,9289533693	0,8573477157
F-Score	0,7290921713	0,934456438	0,8573717529
	Pool 2		
	KNN - crawler	KNN - corpus	KNN - full
Acurácia	0,7297585377	0,9052548499	0,8292676734
Precisão	0,7175800515	0,9096836479	0,8174337755
Recall	0,7167358248	0,8788858379	0,8132554359
F-Score	0,7231885598	0,8918754813	0,8211835065

	Pool 3		
	KNN - crawler	KNN - corpus	KNN - full
Acurácia	0,7289193801	0,9058614845	0,827596187
Precisão	0,7172838435	0,9053707518	0,8180430969
Recall	0,7156217309	0,8855044598	0,8075748864
F-Score	0,7222093501	0,8955673039	0,8174629647

	Pool 1		
	Perceptron - crawler	Perceptron - corpus	Perceptron - full
Acurácia	0,8129621	0,9514134644	0,889457143
Precisão	0,821321	0,947846362	0,879299556
Recall	0,7788476053	0,945740796	0,8829317854
F-Score	0,7955393137	0,9485686493	0,8861824521
	Pool 2		
	Perceptron - crawler	Perceptron - corpus	Perceptron - full
Acurácia	0,7649353685	0,9406324209	0,8658542602
Precisão	0,762403686	0,9351603602	0,8542235837
Recall	0,7391027501	0,9345110296	0,8574725461
F-Score	0,7517972151	0,9375617337	0,8616430202
	Pool 3		
	Perceptron - crawler	Perceptron - corpus	Perceptron - full
Acurácia	0,7222220924	0,9043432273	0,8296609387
Precisão	0,7107055473	0,9071055584	0,8198026138
Recall	0,7071031233	0,8797617011	0,8102930054
F-Score	0,7145826459	0,8918831212	0,819862604

	Pool 1		
	Random Forest - crawler	Random Forest - corpus	Random Forest - full
Acurácia	0,8048604131	0,9544492875	0,8860148745
Precisão	0,8154392615	0,9517451144	0,8837645271
Recall	0,7652156502	0,9479458447	0,8681262545

F-Score	0,7845375122	0,9511864499	0,876979351
	Pool 2		
	Random Forest - crawler	Random Forest - corpus	Random Forest - full
Acurácia	0,7772165214	0,9464014988	0,8723453383
Precisão	0,782122687	0,9406433939	0,869103599
Recall	0,7407226191	0,9416429953	0,8530137395
F-Score	0,7585308817	0,9440162505	0,8625712395
	Pool 3		
	Random Forest - crawler	Random Forest - corpus	Random Forest - full
Acurácia	0,7345048035	0,9041906757	0,8260226905
Precisão	0,7350048751	0,9023318092	0,8154772335
Recall	0,6957796289	0,8852246787	0,8073747175
F-Score	0,7146179712	0,894607167	0,8165922551